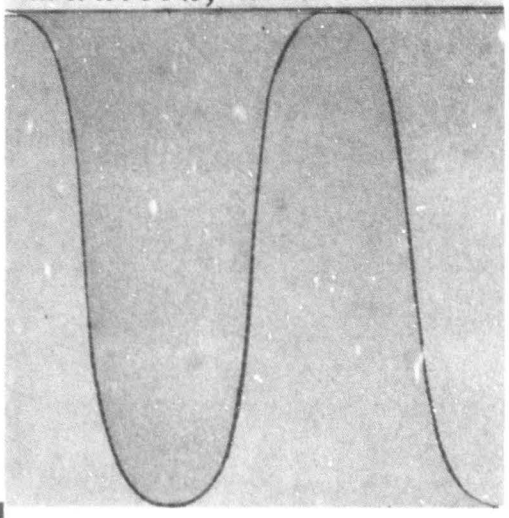
THE UNIVERSITY OF WISCONSIN

*madison, wisconsin*



UNITED STATES ARMY

MATHEMATICS RESEARCH CENTER

MATHEMATICS RESEARCH CENTER, UNITED STATES ARMY
THE UNIVERSITY OF WISCONSIN

Contract No.: DA-31-124-ARO-D-462

# RECENT ADVANCES IN NETWORK FLOWS

T. C. Hu

MRC Technical Summary Report #753
June 1967

Madison, Wisconsin

## ABSTRACT

The present paper surveys important results in the theory of network flows which are not included in the book "Flows in Networks" by Ford and Fulkerson. The survey is divided into three areas: 1. Shortest paths and minimal cost flows, 2. Multi-terminal flows, 3. Multi-commodity flows.

# RECENT ADVANCES IN NETWORK FLOWS

T. C. Hu

This report presents a survey of important papers on network flows, many of which appeared after the publication of the book "Flows in Networks" by Ford and Fulkerson [5]. Still, it will be an extension rather than a duplication of the survey paper by Fulkerson [7]. The areas considered in this survey are: 1. Shortest paths and minimal cost flows, 2. Multi-terminal flows, 3. Multi-commodity flows. Unless stated explicitely to the contrary, it will be assumed that the network has $n$ nodes and $m$ arcs which are either directed or undirected. Associated with every arc, there is an arc capacity $b_{ij}$ which indicates the maximum amount of flow that can pass through the arc that leads from node $N_i$ to node $N_j$. If $x_{ij}$ is the arc flow, then $0 \leq x_{ij} \leq b_{ij}$. A cost $c_{ij}$ may also be associated with this arc, being the cost of shipping one unit of flow along the arc; thus $\Sigma\, c_{ij}\, x_{ij}$ is the total cost of the flow.

1. Shortest paths and minimal cost flows: The minimal cost flow problem is to

$$\min \ \Sigma\, c_{ij}\, x_{ij}$$

$$\text{subject to } \sum_i x_{ij} - \sum_k x_{jk} = \begin{cases} -v & \text{if } j = s \\ 0 & \text{if } j \neq s,\ t \\ v & \text{if } j = t \end{cases}$$

$$\text{and } \ 0 \leq x_{ij} \leq b_{ij}$$

Sponsored by the Mathematics Research Center, United States Army, Madison, Wisconsin, under Contract No.: DA-31-124-ARO-D-462.

where $N_s$ is the source and $N_t$ is the sink. The value $v$ is the required amount of flow from $N_s$ to $N_t$. By setting $v = 1$, the problem can be viewed as a shortest path problem with the $c_{ij}$ interpreted as distances along arcs.

Basically, there are two types of algorithms for the construction of shortest paths; 1) the tree-building algorithm which finds the shortest paths from $N_s$ to all the other nodes, and 2) the matrix algorithm which finds shortest paths between each pair of nodes.

1) When all $c_{ij}$ are nonnegative, Dantzig [3] proposed the following algorithm for the construction of a tree which contains arcs forming shortest paths from $N_s$ to all the other nodes. At the start, no arcs of the network belong to the tree, and arcs of the tree are added one at a time by the following rule. Let $N_i$ be the set of nodes connected by the tree, and $L_{si}$ the shortest distance from $N_s$ to $N_i$. Then let

$$\min_k \min_i (L_{si} + c_{ik}) = L_{sr} + c_{rp} = L_{sp} \qquad k=i+1, i+2, \ldots, n \qquad (1)$$

The arc $rp$ is added to the tree and $N_p$ receives the label $L_{sp}$. The algorithm starts with $N_s$ as the single node of the tree and ends when all nodes are connected by the tree. When there are $q$ nodes in the tree, we need $q(n-q)$ additions and comparisons to add one node to the tree; the total number of additions and comparisons is therefore

$$\sum_{q=1}^{n-1} q(n-q) = \begin{cases} \frac{(2n-1)(n+2)n}{24} & \text{if } n \text{ is even} \\ \frac{2n(n+1)(n-1)}{24} & \text{if } n \text{ is odd} \end{cases}$$

The modification by Karp [17] is to indicate the current shortest distances from $N_s$ to every node on that node whether or not it belongs to the tree. This label is called a permanent label if the node belongs to the tree and a temporary label if the node does not belong to the tree. The temporary

label is calculated just like (1) and the temporary label with minimum distance becomes a permanent label. Each time a node $N_p$ gets a permanent label, we compare each temporary label $L_{sk}$ with $L_{sp} + c_{pk}$, and the minimum of the two becomes the temporary label of that node. Thus at most n additions are needed in computing $L_{sp} + c_{pk}$ $(k = 1 \ldots n)$, at most n comparisons are needed to fix all temporary labels after a permanent label is added, and at most another n comparisons among the temporary labels are needed to decide which one will become permanent. Therefore, no more than 3n operations are involved in getting a permanent label, and a maximum of $3n^2$ operation are needed in the algorithm. This bound is explicitly pointed out by Karp [17] and the algorithm was adopted by Martin [20].

2) When the $c_{ij}$ are allowed to be negative without leading to negative cycles* then usually a matrix-type of algoritnm is used to find the shortest paths between all pairs of nodes. We shall not go through the long history of how the number of operations can be reduced from $n^3 \log_2(n-1)$ to $n^3$ but only state the most recent result. (The interested reader is refered to the papers [4] [12] [22]).

Given the $n \times n$ distance matrix whose entries $c_{ij}$ denote the distances of arcs from $N_i$ to $N_j$, with $c_{ii} = 0$ for all $i = 1, \ldots, n$, and $c_{ij} = \infty$ if there is no arc leading from $N_i$ to $N_j$. We shall use the following operation to change $c_{ij}$

$$c_{ik} := \min (c_{ik}, c_{ij} + c_{jk}) , \qquad (2)$$

where := means to be replaced by. The operation (2) is to be performed for all i and k with the value of j first set at 1. Then (2) is performed

*If the sum of costs of arcs in a cycle is negative, then the cycle is called a negative cycle.

again for all i and k with j set at 2, etc. The operation (2) is completed after j = n. At that time, all entries in the matrix represent the shortest distances. The total number of additions is $n(n-1)^2$ and so is the total number of comparisons.

To keep track of what are the intermediate nodes in the shortest paths, another $n \times n$ matrix is built up the entries of which are calculated along with the calculations of (2). At the start, all entries in the (i, k) positions are set equal to k. Then these entries are changed according to the following rule

$$(i, k) \quad \begin{cases} \text{is set equal to } j \text{ if } c_{ik} > c_{ij} + c_{jk}, \\ \text{remains unchanged if } c_{ik} \leq c_{ij} + c_{jk}. \end{cases} \tag{3}$$

These indices then indicate the intermediate nodes in each shortest path. This method is proposed by [4] [22] [27] and the number of operations is minimal for a completely connected network with arbitarily defined distances. For a large network, where not every pair of nodes is connected by an arc, Hu [13] proposed a decomposition algorithm to save the amount of computation and the storage requirements.

There are two algorithms for solving the minimum cost flow problem, both using a shortest path algorithm as subalgorithm. We shall first discuss the paper by Busacker and Gowen [1]. Their algorithm consists of solving a sequence of shortest path problems where the distances along the arcs are functions of the existing flow in the network. At the start, the arc flows $x_{ij} = 0$.

As before the $b_{ij}$ are the arc capacities, the cost $c_{ij}$ are interpreted as distances along arcs, and the flow is to be sent along the shortest path from $N_s$ to $N_t$. Based on the current flow in the network, new modified cost $c^*_{ij}$ (or modified distances) are defined as follows:

$$\begin{cases} c^*_{ij} = c_{ij} & \text{if } x_{ij} < b_{ij} \\ c^*_{ij} = \infty & \text{if } x_{ij} = b_{ij} \\ c^*_{ij} = -c_{ji} & \text{if } x_{ji} > 0 \end{cases} \tag{4}$$

We then find a shortest path using $c^*_{ij}$ as distances, and ship the maximum amount possible along the new shortest path. This is repeated until the total amount shipped from $N_s$ to $N_t$ is $v$. This can be considered as a dual algorithm since the feasible solution is obtained at the end of the computation.

A primal algorithm suggested by Klein [18] is as follows. First find a flow of $v$ units from $N_s$ to $N_t$ disregarding the costs. This can be done, for example by using the labeling method for finding the maximal flow. Then the modified costs (or modified distances) are defined as in (4). Using these $c^*_{ij}$ as distances, the existence of negative cycles is checked. If there does not exist any negative cycle, then the current flow is optimum; if there exists a negative cycle, a cylce of flow is superimposed on it, the $c^*_{ij}$ are defined and the existence of negative cycles is checked again. The validity of the two algorithms is established by the theorem: A flow is optimum if and only if there are no negative cycles. This theorem can be considered as the central theorem in minimal cost flow. It is stated explicitly in Busacker and Saaty [2] and also implicitly in [16] [21].

2. Multi-terminal flows: In the problem of multi-terminal flows, every pair of nodes can be considered as the source and the sink while all other nodes are regular nodes where the flow is conserved. This is different from the multi-commodity flow problem where flows of many kinds can exist simultaneously in the network. Many results on the multi-terminal flows are given in Gomory and Hu [8] [9], some of which are also described in the 4th chapter of the book by Ford and Fulkerson [5].

In the paper by Gomory and Hu [8], it was shown that for an undirected network i.e. $b_{ij} = b_{ji}$, $n(n-1)/2$ maximal flow values $f_{ij}$ can be found by solving $n-1$ maximum flow problems. If one is interested in finding the maximum flow values in a subset containing $p$ nodes, then $p - 1$ flow computations are sufficient to determine the $p(p-1)/2$ maximum flow values. This extension is also obtained by Gomory and Hu [9] and is contained in the lecture notes of Hu [15]. Gupta [11] showed that the condition $b_{ij} = b_{ji}$ can be weakened by requiring the network to be pseudosymmetric. A network is pseudosymmetric if $f_{ij}$, the number of arcs which leave every node is always equal to $f_{ij}$, the number of arcs which arrive at that node. If such a condition holds in a network, the network can be considered to be s superposition of directed cycles. This then immediately implies $f_{ij} = f_{ji}$. An unsolved problem of the moment is that whether the condition $f_{ij} = f_{ji}$ would imply the network is pseudosymmetric. This problem is proposed by A. J. Hoffman.

3. Multi-commodity flows: In the multi-commodity flow problem, there are many kinds of flows each having its own source and own sink. All the

flows share the same arc capacity, i.e. the sum of all commodity arc flows cannot exceed $b_{ij}$. Two questions are usually asked; 1) to maximize the total sum of different flows, (this is called the maximum multi-commodity flow problem); 2) to prescribe lower bounds on each of the flow values and ask if it is feasible. (This is called the feasibility problem.)

1) For the special case of two-commodity flows in an undirected network, Hu [14] has proved a theorem analogous to the Max Flow Min Cut theorem for one commodity flow and presented a labeling technique for constructing the maximum flows. Rothschild and Whinston [23] have replaced the condition of $b_{ij} = b_{ji}$ by requiring the network to be pseudosymmetric.

For the general maximum multi-commodity flow problem, Ford and Fulkerson [6] first formulated it as a linear programming problem with a constraints - matrix A of m+1 rows and many more columns. The matrix A is the incidence matrix of arcs-versus-chains. A column generating technique was developed which enables one to use the revised simplex method on a $(m + 1) \times (m + 2)$ matrix. The algorithm starts with using any selection of m columns as the basis, and the revised simplex method then produces the prices $\pi$ of each row. Then, the shortest path between one pair of nodes for some commodity is found by using the prices on arcs as distances. This shortest path, which is a column in the matrix A is then introduced into the basis. New prices are then obtained for this new basis after which the iteration is continued. The iteration comes to an end when the shortest path has a distance of 1 or more. If the costs $c_{ij}$ are also involved, then

$c_{ij} - \pi_{ij}$ is used as the distance along an arc, see for example Tomlin [26]. A decomposition scheme for the maximization of multi-commodity flows has recently been proposed by Sakarovitch [24].

2) In the feasibility problem of multi-commodity flows, the approach is basically the same; it can also be formulated as a linear program with a matrix A of $m + 1$ rows. Here the matrix A is an arcs-versus-network incidence matrix where each column represents a feasible network. A column generating technique analogous to the previous one is used. The revised simplex method will produce the prices and a shortest path for each commodity is then found using these prices. The network is then obtained by superimposition of all shortest paths.

The associated synthesis problem of multi-commodity flows is much more complicated. Here we are given a set of lower bounds of flow requirements which are functions of time. It is required to find a minimum cost network which will meet the lower bound requirements in all time periods. Gomory and Hu [10] proposed two algorithms for solving this, both making use of the simplex or dual simplex method together with a row generating technique, the latter itself being a linear program.

Acknowledgement

The author is deeply indebted to Prof. Herman F. Karreman for reading the manuscript and making valuable suggestions for its presentation.

## REFERENCES


1. R. G. Busacker and P. J. Gowen, "A Procedure for Determining a Family of Minimal-cost Network Flow Patterns," O. P. O. Tech. paper 15, 1961.
2. R. G. Busacker and T. L. Saaty, "Finite Graphs and Networks," McGraw-Hill Book Company, 1965, pp. 256.
3. G. B. Dantzig, "On the Shortest Route Through a Network," Manag Sci., Vol. 6, No. 2, Jan. 1960, pp. 187-190.
4. R. W. Floyed, "Algorithm 97: Shortest Path," Communications of the Association for Computing Machinery 5 (1962) 345.
5. L. R. Ford and D. R. Fulkerson, "Flows in Networks," Princeton University Press, 1962.
6. L. R. Ford and D. R. Fulkerson, "A Suggested Computation for Maximal Multi-Commodity Network Flows," Management Science 5, 1958, pp. 97-101.
7. D. R. Fulkerson, "Flow Networks and Combinatorial Operations Research," Am. Math. Monthly, 73 (1966), pp. 115-138.
8. R. E. Gomory and T. C. Hu, "Multi-terminal Network Flows," J. of SIAM. Dec. 1961.
9. R. E. Gomory and T. C. Hu, "An application of Generalized Linear Programming to Network Flows," J. of SIAM, 10 (1962), pp. 260-283.
10. R. E. Gomory and T. C. Hu, "Synthesis of a Communication Network," J. of SIAM. June (1964), pp. 348-269.

11. R. P. Gupta, "On Flows in Pseudosymmetric Networks," J. of SIAM. March 1966, pp. 215-226.

12. T. C. Hu, "Revised Matrix Algorithms for Shortest Paths in a Network," J. of SIAM. Jan. 1967, pp. 207-218.

13. T. C. Hu, "A Decomposition Algorithm for Shortest Paths in a Network," IBM Res. Report RC-1562. Feb. 1966.

14. T. C. Hu, "Multi-Commodity Network Flows," J. of ORSA 11 (1963).

15. T. C. Hu, Lecture Notes, University of Wisconsin, Madison.

16. W. S. Jewell, "Ophmum Flow Through Networks" Interim Tech. Report. No. 8. M.I.T. 1958.

17. R. M. Karp, Private Communication, also contained in [15].

18. M. Klein, "A Primal Method for Minimum Cost Flows with Application to the Assignment and Transportation Problems," Tech. Report No. 32, O. R. Group, Columbia University, June, 1966.

19. A. H. Land and S. Stairs, "The Extension of the Cascade Algorithms to Larger Graphs," LSE-TNT-20. London School of Economics, 1965.

20. C. V. Martin, Civil En. Dept. Report, M.I.T., 1963.

21. G. J. Minty, "Monotone Networks," Proc. Roy. Soc., London, Ser. A, 257. (1960), pp. 194-212.

22. J. D. Murchland, "A New Method for Finding All Elementary Paths in a Complete Directed Graph," LSE-TNT-22. London School of Economics. Oct., 1965.

23. B. Rothschild and A. Whinston, "On Two Commodity Network Flows," J. or ORSA. May-June 1966, pp. 377-388.

24. M. Sakarovitch, "The Multi-Commodity Maximum Flow Problem," ORC 66-25, University of California (Berkeley), 1966.

25. A. Shimbel, "Applications of Matrix Algebra to Communication Nets," Bulletin of Math Biophysics 13 (1951).

26. J. A. Tomlin, "Minimum-cost Multi-commodity Network Flows," J. of ORSA, Jan. - Feb. 1966, pp. 45-51.

27. S. Warshall, "A Theorem on Boolean Matrices," J. ACM 9 (1962).